6L-08

スーパーコンピュータを用いた SPIRAL コードジェネレータの性能評価

北井 成哉† 片桐 孝洋‡ 永井 亨‡ 荻野 正雄‡

名古屋大学 工学部電気電子・情報工学科† 名古屋大学 情報基盤センター‡

1. はじめに

近年の計算機は，計算能力を向上させるためにメモリの階層化，CPU のメニーコア化，GPGPU の利用といったシステムの複雑化，多様化が進んでいる．このような変化は計算能力の大幅な向上をもたらした一方で，各環境に合わせて効率的なソフトウェアのチューニングを行うためのコストも増大させている．よって，ソフトウェアの性能の移植性の向上が必要となってきており，その取り組みの一つに，自動チューニング技術[1]の適用による，コード自動生成がある．

SPIRAL[2]はプログラム自動生成システムであり，離散フーリエ変換等のコードを最適化生成する機能を持つ．

本研究では，SPIRAL により自動生成されたコードの性能評価を，スーパーコンピュータにより行うことを目的とする．

2. SPIRAL コードジェネレータ

SPIRAL は離散フーリエ変換や，SPIRAL で提供する抽象度で記載できる処理に対する変換を行う，C 言語，Fortran の最適化したコードを自動生成するシステムである．コードの最適化は SIMD 命令の適用や，ループアンローリング等を行うパラメータの指定により行われる．指定された情報は，コードの生成手順の中で順次適用される．

まず，SPIRAL は離散フーリエ変換，離散コサイン変換等のアルゴリズムの生成機能を備えている．変換のサイズを指定すると，サイズに応じたアルゴリズムが生成される．このアルゴリズムは，より小さい演算の木(rule tree)に分解され，ここでいくつかの生成可能な木からより性能の良いものを選択する．

Performance evaluation of SPIRAL code generator using supercomputer

† Naruya Kitai, Electrical and Electronic Engineering and Information Engineering, School of engineering, Nagoya University

‡ Takahiro Katagiri, Toru Nagai, Masao Ogino, Information Technology Center, Nagoya University

rule tree は，データの流れの木 (data flow graph) に変換され，ループ構造(Σ-SPL)が作られた後に，抽象構文木が生成される．

最後に，抽象構文木は構文解析されて，C 言語，Fortran のコードが生成される．

このように SPIRAL のコード生成はアルゴリズムの選択から始まっており，コード最適化はアルゴリズム，ハードウェアの面からそれぞれ独立して調整することができる．アルゴリズムの部分には，rule tree の選択やループアンローリングの指定，ハードウェアの部分には SIMD 命令の指定などが当てはまる．

3. 予備実験

SPIRAL により自動生成された実離散フーリエ変換プログラムの性能評価を行う．最適化のパラメータの指定はせず，2 種類の rule tree（最左導出，中間の導出）を選択してそれぞれの場合の評価を行う．この場合，プログラムは逐次実行され，明示的な SIMD 命令も使用されない．

また，同じ条件下で，自動チューニング機能付き FFT ライブラリ FFTW を用いた場合の評価も行い，SPIRAL のコード生成の性能を比較する．

計算機環境として，「京」コンピュータ型のスーパーコンピュータである，名古屋大学情報基盤センター設置の Fujitsu PRIMEHPC FX100 を利用する．コンパイラは，富士通社製の C++コンパイラ fccpx Version 2.0.0 を利用した．自動生成コードをコンパイルする際のコンパイラオプションは， -Kfast -Nfjcex を使用した．

4. 評価結果

表 1，表 2，表 3 に，SPIRAL で最左導出を選択した場合，SPIRAL で中間の導出を選択した場合，および，FFTW を用いた場合の実行時間と実行性能をそれぞれ示す．

表 1，表 2 の結果から rule tree の選択によってプログラムの実行性能が大きく変わることが分かる．DFT のサイズが 512 の場合，中間の導出のほうが最左導出のものより 3 倍以上効率が良いことが分かる．

表 1 SPIRAL で最左導出を用いた場合

DFT のサイズ	実行時間（μs）	実行性能（GFlops）
4	0.025	0.800
8	0.025	2.400
16	2.450	0.065
32	0.125	3.200
64	0.325	2.953
128	3.525	0.635
256	5.875	0.871
512	13.375	0.861
1024	17.925	1.428

表 2 SPIRAL で中間の導出を用いた場合

DFT のサイズ	実行時間（μs）	実行性能（GFlops）
4	0.025	0.800
8	0.025	2.400
16	0.050	3.200
32	0.100	4.000
64	0.275	3.491
128	2.050	1.093
256	2.625	1.950
512	3.525	3.268
1024	17.425	1.469

表 3 FFTW を用いた場合

DFT のサイズ	実行時間（μs）	実行性能（GFlops）
4	0.175	0.114
8	0.075	0.800
16	0.150	1.067
32	0.300	1.333
64	0.525	1.829
128	0.925	2.422
256	1.975	2.592
512	4.225	2.727
1024	8.875	2.885

一方，表 2，表 3 を比べると SPIRAL で自動生成されたプログラムの実行効率が FFTW のものを上回る場合があることが分かる．特に DFT のサイズが 64 以下の時，SPIRAL コードのほうが FFTW よりも性能が高く，サイズが 4 のときは SPIRAL コードのほうが約 7 倍，FFTW より高速である．この理由は，SPIRAL では基数ごとに専用コードを生成するため，小さいサイズの実行では，レジスタブロッキング等が促進されることが考えられるが，詳細な解析は今後の課題である．

5. まとめ

本研究では，プログラム自動生成システム SPIRAL による，離散フーリエ変換等のコードを最適化生成する機能の評価を，SPIRAL により自動生成されたコードを性能評価することで行った．性能評価には，「京」コンピュータ型のスーパーコンピュータである，名古屋大学設置の Fujitsu PRIMEHPC FX100 を利用した。

予備実験の結果から，SPIRAL により自動生成されたプログラムの実行性能は rule tree の選択により大きく変化することが分かった．また，rule tree の選択によっては fftw の性能を上回る場合があることが分かった．

ただし，今回の実験結果は逐次実行のものであり，離散フーリエ変換のサイズも比較的小さいものである．

今後の課題として，並列実行の場合，大きいサイズの場合における性能評価が必要である．さらに，効率の良い rule tree の選択方法の調査と実装も必要である．

謝辞

本研究の一部は，JSPS 科研費 17H05290，16H02823 の助成による．

参考文献

[1] Takahiro Katagiri and Daisuke Takahashi, “Japanese Auto-tuning Research: Auto-tuning Languages and FFT”, Proceedings of the IEEE, Volume 106, Issue 11, pp. 2056-2067 (2018)

[2] Franz Franchetti, Tze-Meng Low, Thom Popovici, Richard Veras, Daniele G. Spampinato, Jeremy Johnson, Markus Püschel, James C. Hoe and José M. F. Moura, “SPIRAL: Extreme Performance Portability” in Proc. IEEE, special issue on “From High Level Specification to High Performance Code”, Vol. 106, No. 11 (2018)